MARS CLIMATE DATABASE v5.2 USER MANUAL

(ESTEC Contract 11369/95/NL/JG
"Mars Climate Database and Physical Models")

(CNES Contract "Base de données climatique martienne")

E. Millour, F. Forget (LMD, Paris)
S. R. Lewis (The Open University, Milton Keynes)

February 2015

Abstract

This document is the User Manual for version 5.2 of the Mars Climate Database (MCD) developed by LMD (Paris), AOPP (Oxford), Dept. Physics & Astronomy (The Open University) and IAA (Granada) with the support of the European Space Agency and the Centre National d'Etudes Spatiales. This is a database of atmospheric statistics compiled from Global Climate Model (GCM) numerical simulations of the Martian atmosphere. This document replaces previous documents which describe versions 5.1, 5.0, 4.x, 3, 2, and 1. and includes a thorough description of the access software provided to extract and postprocess data from the database.

The database extends up to exobase (the top of the thremosphere, roughly at 300 km in altitude); in addition to statistics on temperature, wind, pressure, radiative fluxes, it provides data such as atmospheric composition (including dust, water vapor and ice content) and make use of "dust and Extreme UltraViolet (EUV) scenarios" to represent the variation of dust in the atmosphere and solar EUV conditions. The data sets of MCD version 5.1 include scenarios for Mars Years 24 to 31, topped by corresponding realistic EUV day to day forcing. Linear interpolation in time of datafiles data is used to reconstruct variables at user-specified time of day and solar longitude and various kind of vertical coordinates may be specified as input. The MCD access software, the Fortran routine "call_mcd" (which is the same interface developped for MCD versions 5.x) includes a "high resolution mode" via postprocessing of MCD data which combines high resolution MOLA (32 pixels/degree) topography and atmospheric mass correction from Viking Lander 1 pressure records. Examples of interfaces for users interested in calling subroutine "call_mcd" from C or C++ programs or IDL, Matlab and Scilab softwares are also given.

Two seperate light tools, "pres0", which yields high resolution surface pressure and "heights", which converts various vertical coordinates are also provided.

For descriptions of the contents and structure of the datafiles, details on the dust distribution scenarios, a description of the variability models, and of the high resolution postprocessing, see the **Detailed Design Document**.

Contents

1 Introduction

The Mars Climate Database (MCD) is a database of atmospheric statistics compiled from state-of-the art Global Climate Model (GCM) simulations of the Martian atmosphere.

The GCM computes in 3D the atmospheric circulation and climate taking into account radiative transfer through the gaseous atmospheres and the dust and ice aerosols, includes a representation of the CO_2 ice condensation and sublimation on the ground and in the atmosphere, simulates the water cycle (with modelling of cloud microphysics), the dust multisize particle transport, the atmospheric composition controlled by the photochemistry and the local non-condensible gas enrichment and depletion induced by CO_2 condensation and sublimation, and has been extended into the thermosphere and to model ionospheric processes (due to chemistry).

The models used to compile the statistics have been extensively validated using available observational data and represent the current best knowledge of the state of the Martian atmosphere given the observations and the physical laws which govern the atmospheric circulation and surface conditions on the planet.

The Mars Climate Database access software moreover includes several post-processing schemes to better represent and account for the Martian environment variability and accurately compute surface pressure and atmospheric temperature at high spatial resolution.

The MCD is freely available to all on request. A light (basic) version may also be accessed to make plots and figures using the interactive server on our WWW site at:
`http://www-mars.lmd.jussieu.fr`

1.1 Available Data.

The MCD contains several statistics on simulated data stored on a $5.625° \times 3.75°$ longitude–latitude grid from the surface up to an approximate altitude of 300 km: temperature, wind, density, pressure, radiative fluxes, atmosphere composition and gases concentration, CO_2 ice surface layer, statistics on convection, etc...

Fields are averaged and stored 12 times a day, for 12 Martian "months" to give a **comprehensive representation of the annual and diurnal cycles**. Each month covers 30° in solar longitude (L_s), and is typically 50-70 days long. In other words, at every grid-point, the database contains 12 "typical" days, one for each month. In addition, information on the variability of the data within one month and the day to day oscillations are also stored in the database. Software tools are provided to reconstruct and synthetize this variability (section 5.1).

1.2 Database scenarios.

Eight combinations of dust and solar scenarios are provided because these are the two forcings that are highly variable from year to year.

- On the one hand, the solar conditions describe variations in the Extreme UV input which control the heating of the atmosphere above 120 km, which typically varies on a 11 years cycle. Depending on the scenarios, either fixed (i.e. constant over time) solar maximum average and/or minimum conditions are provided, or varying (i.e. changing from day to day, as observed) realistic solar EUV.

- On the other hand, the major factor which governs the variability in the Martian atmosphere is the amount and distribution of suspended dust. Because of this variability, and since even for a given year the details of the dust distribution and optical properties can be uncertain, various model integrations were carried out for the database assuming different "dust scenarios", i.e. prescribing various amount of airborne dust in the simulated atmosphere. 5 kinds of dust scenarios are proposed :

1. The **“Climatology” (clim)** scenario, which is a simulations ran using the latest version of the LMD Global Climate Model (GCM) forced by a dust distribution reconstructed from observations over Mars Years[1] 24 to 31, and thus representative of a standard (i.e.: devoided of a planet-encircling global dust storm) Martian year. This “Climatology” scenario is provided with 3 solar EUV conditions: solar min, solar ave, solar max.
2. The **cold** scenario corresponds to an extremely clear atmosphere (“Low dust scenario”); where the dust opacity at a given location is set to be the minimum observed over Mars years 24-31, further decreased by 50%, and in which the readiative effect of clouds is neglected, topped with a solar EUV minimum thermospheric forcing.
3. The **warm** scenario corresponds to “dusty atmosphere” conditions, but nonetheless non-dust storm conditions (the dust opacity at a given location is set to the maximum observed, unless during a global dust storm, further increased by 50%), topped with a solar maximum thermosphere.
4. The **dust storm** scenario represents Mars during a global dust storm (dust opacity set to $\tau = 5$ at all times and over the whole planet). Moreover the dust optical properties are for this case set to represent “darker dust” than nominal (in practice for these runs, Ockert-Bell et al. dust properties were used, rather than the more recently derived Wolff et al. ones). This scenario is only provided for when such storms are likely to happen, during northern fall and winter (Ls=180-360), but with 3 cases of solar EUV inputs: solar min, solar ave, solar max.
5. The **Mars Years** scenarios (MY24, MY25, ..., MY30, MY31) correspond to our best representation of these specific years, both in terms of daily atmospheric dust loading and daily solar EUV input.

The “cold” and “warm” annual scenarios are provided to bracket the possible global conditions on Mars outside global dust storms which are known to be highly variable locally and from year to year. The individual Mars Year scenarios are provided for users interested in investigating the atmospheric states as they effectively occured over the last (Earth) decades. Overall, this leads to a total of 16 different available scenarios (since there are 3 EUV cases for “Climatology” and “dust storm” scenarios).

The user is referred to the Detailed Design Document for further information on specific aspects of these dust scenarios.

1.3 High resolution mode

The Mars Climate Database has been compiled from the output of a general circulation model in which the topography is very smoothed because of its low resolution. In addition, the pressure variations due to the CO2 cycle (condensation of atmospheric CO2 in the polar caps) that is computed by the model is only based on the simulation of the actual physical processes. The polar cap physical properties have been tuned to somewhat reproduce the observations, but no correction was added.

As of version 4.2, access software includes a “high resolution” mode which combines high resolution (32 pixels/degree) MOLA topography and the smoothed Viking Lander 1 pressure records (used as a reference to correct the atmospheric mass) with the MCD surface pressure in order to compute surface pressure as accurately as possible. The latter is then used to reconstruct vertical pressure levels and hence, within the restrictions of the procedure, yield high resolution values of atmospheric variables.

The procedure by which high accuracy surface pressure is computed is also implemented as a light and autonomous tool, `pres0` (see section 11).

[1]This widely used numbering of martian years follows the calendar proposed by R. Todd Clancy (Clancy et al., *Journal of Geophys. Res* 105, p 9553, 2000) which begins on April 11, 1955 (Ls=0°)

2 Contents of the Mars Climate database

The contents of each subdirectory of the MCD distribution are summarized here.

`docs` This directory contains files in pdf formats which can be used to print further copies of the documentation:

- The User Manual (`user_manual.pdf`) of the database; this document.
- Detailed Design Document (`detailed_design.pdf` of the database v5.2
- pdf versions of the scientific reference articles describing various aspects of the Global Climate Model used to compile it are also provided.

`mcd` This directory contains Fortran source code for the climate database access softwares (see section 5, and the `README` file in the directory): the CALL_MCD subroutine, and a test program.
Subdirectory `testcase` contains a simple tool to test the results from the software after installation.
Subdirectory `pres0` contains an autonomous tool to compute surface pressure in the context of high resolution topography (see section 3.2).
Subdirectories `idl`, `matlab`, `scilab`, `c_interface` and `python` contain examples of interfaces of the Fortran subroutine with other languages and softwares.

`data` The full MCD datasets derived from model runs. Essential and common datafiles (`VL1.ls`, `VL3.ls`, `mgm1025`, `mola32.nc`, `mountain.nc` and `ps_clim.nc`) are in this directory. Individual dust scenarios datafiles are in corresponding subdirectories (`clim_aveEUV` for the baseline climatology dust scenario with average Extreme UV (EUV) input, `clim_minEUV` for the baseline climatology dust scenario with minimum EUV input, `clim_maxEUV` for the baseline climatology dust scenario with maximum EUV input, `cold` for the cold scenario, `warm` for the warm scenario and `strm` for the dust storm scenarios). Additional scenarios representing the last 8 Mars years should also be placed in corresponding `MY24`, `MY25`, `MY26`, `MY27`, `MY28`, `MY29`, `MY30` and `MY31` subdirectories.

3 Installation

3.1 System and Software Requirements

- The MCD is primarily designed to operate in the **Unix/Linux** environment on a PC or a workstation. Access software is written in **Fortran** (and uses some Fortran90 extentions and intrinsics), for which a Fortran compiler is needed.

  Because the NetCDF libraries (see below) are also available under **Windows** systems, several users have succefully compiled and used the access software as under **Unix/Linux**. Nevertheless, it has not been fully tested by our team but we can confirm that it compiles fine under the **Cygwin** environment on Windows using the GNU gfortran compiler.

- The data in the MCD is written as Network Common Data Form (**NetCDF**) files. The NetCDF library is freely available from the Unidata web site :

  `http://www.unidata.ucar.edu/software/netcdf/`

  NetCDF works on most current operating systems, including:
  - AIX
  - HPUX
  - IRIX
  - Linux
  - MacOS X

  - OSF1
  - SunOS-4, Solaris (Sparc and i386)
  - MS Windows

3.2 Installing the MCD

1. Create a working directory, e.g. `mars`, on a disk where you wish to use the database.

2. Extract (untar) the contents of the MCD distribution there.

3. With the installation of the minimal baseline MCD distribution, only the "climatology" dust scenario with an average Extreme UV solar input is provided (corresponding datafiles are in the `clim_aveEUV` subdirectory of the `data` directory). The other dust scenarios (`clim_minEUV`, `clim_maxEUV`, `cold`, `warm`, `strm`, `MY24`, `MY25`, `MY26`, `MY27`, `MY28`, `MY29`, `MY30`, `MY31`) are provided separately and should likewise be added as subdirectories in the `data` directory. A full installation (all 16 scenarios) of the MCD takes about 26 Gb of disk space.

4. In the working directory (i.e. where you will run the software) it is convenient to set up a `MCD_DATA` symbolic link[2] to point to the MCD `data` directory, wherever it has been stored:

   ```
   ln -s /full/path/to/mcd/data MCD_DATA
   ```

   N.B In the `call_mcd` subroutine , the path to the directory can be given as an input using the `dset` argument (e.g. `dset='/full/path/to/mcd/data/'`) although by default the subroutine will use `MCD_DATA/` if `dset` is not initialized or set to `' '`.

5. If NetCDF is not available on your system, you must install the NetCDF library following the instructions given on their www site (see above). For this purpose, you have the choice either to build and install the NetCDF package from source, or use prebuilt binary releases if they are available for your platform (check the "Frequently Asked Question").

   Ideally, you can install the full NetCDF package as recommanded on the web site. In practice, the minimum you need to run the access software are only two files : an include file named `netcdf.inc` and a library file named `libnetcdf.a` (for older ,i.e. pre-version 4 NetCDF; the more recent versions separate Fortran and underlying C libraries as `libnetcdff.a` and `libnetcdf.a`). The version of the files depends of the machine and of the compiler. An easy way to get them is to go to the unidata netcdf web page, click on *precompiled binaries*, download a compressed file corresponding to your operating system, uncompress the file. You'll find the `netcdf.inc` in the `include` directory, and the `libnetcdf.a` in the `lib` directory. You'll need to provide the path to these two files when compiling applications (see the `compile` files in the database). Unfortunately, this does not always work : if your compiler doesn't work with the precompiled library, you will have to recompile Netcdf with it, and follow the web site instructions.

4 Ways to access the database

There are three main ways of accessing data from the MCD which have been implemented to date.

Firstly, if you know Fortran, the **best way** to retrieve environmental data from the Mars climate database at any given locations and times is to use the **subroutine mode** of the software supplied with the Mars Climate Database. In practice, one only has to call a main subroutine named `call_mcd` from within any program written in Fortran. A simple example of such a program

[2]This "symbolic link" strategy unfortunately *does not work* with Windows where the full path to the data directory must be used.

(`test_mcd.F`), which can be easily modified, is provided. This mode was developed with particular attention to trajectory simulation applications, but it can also be used for most other purposes. Further information on the use of `call_mcd` are available below.

Secondly, users who prefer using **IDL**, **Matlab** or **Scilab** software, or who program in **C**, **C++** and/or **Python** will find examples of how to interface their favorite tools with the Fortran subroutine `call_mcd` in corresponding subdirectories (see sections 6 to 10). Note that some of the provided examples are quite straightforward and brute force approaches; fancier and more efficient couplings with the MCD are clearly possible.

Thirdly, it is possible to access the database directly from within any program written in any high level language or software which can read NetCDF files. This gives some flexibility for particular applications (e.g. when one wants to handle global fields), although it does demand a much greater understanding of how the database contents are organised (see the MCD Detailed Design Document for a description), as well as of how the variability models, if these are required, should be used. The Fortran subroutines used by the `call_mcd` routine (included in file `call_mcd.F`) illustrate how to open and read the database files.

If you are interested in inspecting/plotting mean and standard deviation data from the raw database datafiles, you can install the Grid Analysis and Display System (**GrADS**) or **Ferret** which are fine free softwares for displaying graphical output from geophysical datasets. GrADS and Ferret can read NetCDF files and display their contents using a few easy instructions. Both works on **Unix/Linux** and **Windows** environments, and can be downloaded from their respective World Wide Web servers:
`http://grads.iges.org/grads`
`http://ferret.pmel.noaa.gov/Ferret`

There are also freely available plotting tools such as **ncview**, **ncBrowse** or **Panoply** which can be used to visualize NetCDF files. IDL users can also read and display the raw datafiles using the "read_ncdf.pro" IDL function available on many websites.

Please note that the vertical coordinate in the datafiles are terrain-following "sigma-pressure" hybrid coordinates. The "altitude" coordinate given in the datafiles is merely an approximation of the real altitude of the data. The Fortran access software calculates heights accurately by integrating the hydrostatic equation directly on the hybrid coordinates.

5 Using the CALL_MCD subroutine

5.1 What is the CALL_MCD subroutine ?

The purpose of the `call_mcd` subroutine is to extract and compute meteorological variables useful for atmospheric trajectory computations as well as scientific studies. Data which may thus be obtained includes:

- Atmospheric and surface pressure
- Atmospheric and surface temperature
- Density
- Radiative fluxes (Solar and thermal IR) on the ground and at the top of the atmosphere
- Wind speed defined by two components the meridional wind (positive when oriented from south to north) the zonal wind (positive when oriented from west to east)
- Vertical wind
- The main atmospheric composition : CO_2, N_2, Ar, CO, O volume mixing ratio
- Mixing ratios of trace gases such as O_2, O_3, H, H_2
- Electron density (up to ∼200km)
- Water vapour and water ice content
- Column values of all species

- Dust aerosol opacity and distribution
- Air viscosity, heat capacity and C_p/C_v ratio
- Sensible heat flux and surface stress
- Key convective plantery boundary layer (PBL) variables (maximum convective updraft and downdraft velocities in the PBL, PBL height, convective vertical wind variance and convective eddy vertical heat flux)

The Fortran subroutine `call_mcd` retrieves database data at any date (Earth date or Mars solar longitude and local time) and at any point in space defined by latitude and east longitude and a vertical coordinate which can be a distance from the planet center, an altitude (above local surface or above reference aeroid), or a pressure level. The returned values of meteorological variables are computed by interpolation in space, time of day and month from data stored in the MCD. As of version 4.2 of the MCD an additional high resolution interpolation procedure (which uses 32 pixels/degree MOLA topography) has been implemented to simulate local pressure and density as accurately as possible.

Above the top level of the database density and pressure are estimated by integration of the hydrostatic equation assuming a constant temperature.

For these variables, the subroutine delivers mean values and, if requested, adds a different type of perturbation to density, pressure, temperature and winds. The available types of perturbation are :

- Small scale perturbations due to the upward propagation of gravity waves for any altitudes (there is no small scale perturbation for surface pressure).
- Large scale perturbations due to the motion of baroclinic weather systems and other transient waves. These perturbations are correlated in longitude and altitude, and are reconstructed from the actual system predicted by the model.
- Perturbations equal to n times the RMS day to day variation for all variables.

The two first types of perturbation have a random component. A comprehensive explanation of the perturbations is included in the Detailed Design Document[3]. See also section 5.5.1 for some discussion and recommendations on the best way to use perturbations to generate realisticly perturbed atmospheres.

[3] In summary the perturbations are calculated as follow :

- The **gravity wave perturbation** of a meteorological variable is calculated by considering vertical displacements of the form
$$\delta z = \delta h \sin\left(\frac{2\pi z}{\lambda} + \phi_0\right) \tag{1}$$
where λ is a characteristic vertical wavelength for the gravity wave and ϕ_0 is a randomly generated surface phase angle. surface angle. δz is the amplitude of the wave depending on the height z. δh is the sub-grid scale surface roughness (the variability on scales smaller than the explicit database resolution) and is a function of location on the Martian surface. If z is higher than 100 km, the amplitude of the wave is taken to be equal to the amplitude at 100 km.The amplitude of the wave is limited to $\lambda/2/\pi$ to saturate the amplitude of the perturbation when it becomes statically unstable.
- The **large scale perturbation** in the Mars Climate Database is represented using a technique that is widely used in meteorological data analysis, namely, Empirical Orthogonal Function (EOF) analysis. A two-dimensional, multivariate EOF of the main atmospheric variables (surface pressure, atmospheric temperature and wind components) is used which describes correlations in the model variability as a function of both height and longitude. 200 EOFs have been retained in the series in order to reproduce the variability of the original fields. Above the top level of the database, the perturbation represents a constant percentage of the mean value this percentage is equal to those at the top of the database.
- For the last type of perturbation (i.e. n times the standard deviation), the standard deviation is interpolated from the day to day RMS variabilities stored in the database. If the user works with pressure as the vertical coordinate, then the added variabilities are pressure-wise, and altitude-wise otherwise. Above the top level of the database, the standard deviation represents a constant percentage of the mean value ; this percentage is equal to those at the top of the database.

5.2 Software Package

The `call_mcd` subroutine is in directory `mcd`. This directory includes:

- `README`, a short text file which summarizes the information contained here.
- File `call_mcd.F`, which contains the CALL_MCD main subroutine and most of the subroutines and functions it calls.
- the *include* file `constants_mcd.inc` used by `call_mcd` and subsidiary routines.
- File `test_mcd.F` which contains a simple and straightforward illustration of a program using `call_mcd` and `julian`.
- The `compile` file which contains an example of the (Unix) command line to compile the subroutine and program.
- The file `test_mcd.def` contains a list of input data for `test_mcd` (see section 5.3).
- A subdirectory `testcase` containing test cases used to test the accuracy of your installation of the database .
- the `julian.F` file, which contains a subroutine which computes the Julian date corresponding to a given calendar date.
- the `heights.F` file which contains the subroutines necessary to convert distances expressed as distance to planet center, height above areoid and height above local surface. Given any of these, this routine computes the other two, either at GCM grid resolution or using high (1/32 degree) resolution (see section 12). `call_mcd` does these conversions using these routines so users need not use it. These routines are nonetheless kept seperate from the main file `call_mcd.F` for specialists who might want to use it seperately.
- subdirectory `pres0` which contains the `pres0` tool (see section 11).
- subdirectories `idl`, `matlab`, `scilab`, `c_interfaces` and `python` which contain examples of interfaces.

5.3 Compiling and running CALL_MCD

A simple program using the `call_mcd` subroutine (as well as the complementary subroutine `julian`) called `test_mcd` is provided in the `mcd` directory. You can easily modify it or use part of this code for your own purpose. To compile the program, edit the `compile` file and make the necessary changes (e.g.: compiler name, path to NetCDF library and include file, see comments in the script). Then, for instance, to compile and run *test_mcd*, type[4]:

```
> compile
> test_mcd
```

Then, just answer the questions...

Alternatively you can edit the file `test_mcd.def` and redirect it to `test_mcd`:

```
> test_mcd < test_mcd.def
```

If the program return an error, see the list of return error codes in table 2 to identify the problem.

In the `mcd/testcase` sub-directory, a tool to test that `call_mcd` is running accurately on your computer (using `test_mcd`) is provided. Please read `mcd/testcase/README` for further information.

If your machine runs under **Windows** you have 2 solutions:

[4] In the examples given here, the > at the begining of command lines is the Unix session command prompt.

1. Install a Unix environment emulator for Windows. The most popular is **Cygwin**, which you can download from `http://www.cygwin.com`. This emulator includes most Unix features and softwares. NetCDF libraries can be built on Cygwin as easely as on other Unix systems. As far as a few tests have shown, requirements and steps necessary to install the Mars Climate Database and use the provided access software under Cygwin are in fact the same as those for 'standard' **Unix** systems.

2. Port the Mars Climate Database to Windows. We have not fully tested that possibility, but several users have done so successfully. NetCDF librairies for Windows can be dowloaded from the NetCDF official website at: `http://www.unidata.ucar.edu/packages/netcdf` Note that, with Windows, the "symbolic link strategy" to the Mars Climate Database `data` directory described in section 3.2 will not work; the 'true' path to that directory must be used in the Fortran routines and programs (see variable `dset` in the description of `call_mcd` arguments in section 5.4).

5.4 CALL_MCD input and output arguments

A Fortran call to subroutine `call_mcd` should look something like:

```
   call_mcd(zkey,xz,xlon,xlat,hireskey,
 &          datekey,xdate,localtime,dset,scena,
 &          perturkey,seedin,gwlength,extvarkeys,
 &          pres,dens,temp,zonwind,merwind,
 &          meanvar,extvars,seedout,ier)
```

All the input arguments (ie: values which must be set before calling the routine and which are not altered by it) are described in table 1. Outputs are described in table 2. The argument list (and types) is identical to what it was in MCDv5.0. MCDv 4.3 users should note that the only change in the argument list is with input argument `extvarkeys` which is now an array (see table 1).

As `call_mcd` runs, it writes informational (and error) messages to standard output. Users who wish to run `call_mcd` silently (i.e. without any messages sent to standard output) should edit file `constants_mcd.inc` and change the value of parameter `output_messages` to `.false.`
As of version 4.3 the "standard output" unit number which will be used by `call_mcd` is set to the value of the `out` parameter, also defined in file `constants_mcd.inc`. The default value of `out` is 6, which is the standard value preconnected to the screen (on most systems). Setting `out` to any other positive integer value n (except 5 which is usually preconnected to standard input) will send messages to the corresponding file. It is thus advised to open the corresponding file (using Fortran command `open(unit=`n`,file="myfilename"))` prior to any call to `call_mcd` otherwise default file `fort.`n will be used (this behaviour is possibly system-dependent).

Table 1: **CALL_MCD Input arguments**

Name	**Type**	**Description**
`zkey`	integer	Flag to set the type of vertical coordinate `xz` is given as: 1: xz is the radial distance from the center of the planet (m). 2: xz is the altitude above the Martian zero datum (Mars geoid or "areoid"), in meters. 3: xz is the altitude above the local surface (m). 4: xz is the pressure level (Pa). 5: xz is the altitude above reference radius ($3.396\,10^6$ m) (m). **N.B.** The zero elevation is defined as the gravitational equipotential surface whose average value at the equator is equal to the mean radius as determined by MOLA. For more informations, see `http://ltpwww.gsfc.nasa.gov/tharsis/mola.html`. Depending on the value of flag `hireskey`, references to areoid and topography are with respect to GCM grid or high resolution MOLA data.
`xz`	real	Vertical coordinate of the requested point. Its exact definition depends on the value of input argument `zkey`.
`xlon`	real	East Longitude (planetocentric), in degrees.
`xlat`	real	Latitude (planetocentric), in degrees.
`hireskey`	integer	Flag to set the resolution at which data retrieval and postprocessing should be done: 0: Interpolate data from GCM 5.625 deg $\times 3.75$ deg grid. 1: Use high resolution (32 pix/deg.) MOLA topography and areoid, as well as some internal post-processing scheme to reconstruct data (see section 1.3).
`datekey`	integer	Flag to set the way dates (`xdate` and `localtime`) should be interpreted: 0: "Earth time"; `xdate` is given in Julian days. With `datekey=0`, the `localtime` argument, although unused, **must be set to zero**. 1: "Mars time"; `xdate` is the value of the solar longitude Ls and `localtime` is the local true solar time at longitude `lon`, given in martian hours.

Table 1: **CALL_MCD Input arguments (continued)**

`xdate`	double precision (`REAL*8`)	• if `datekey=0`: the Julian date. • if `datekey=1`: the solar longitude Ls (in degrees), $Ls \in [0:360]$ **N.B.** The subroutine `julian.F` can be used to compute the Julian date corresponding to a given calendar date (day, month, year, hours, minutes, seconds) on Earth. Note that this date essentially matters in order to compute the corresponding solar longitude Ls of Mars (and that a different Earth date leading to same Ls will yield identical results).
`localtime`	real	Local true solar time at longitude `lon`, in martian hours. Should only be specified if `datekey=1` and must be set to zero if `datekey=0`. **N.B.** Local true solar time is such that the sun is highest in the sky at noon. A martian hour is defined as $1/24^{\text{th}}$ of a sol (a martian day, which is 88775.245 s long).
`dset`	character string dim($*$)	Path to the directory where the datafiles are to be found. `dset` may be of any size. If `dset` is an empty string, then the path to the datasets is set the default path `MCD_DATA/`. **N.B.** the given path must end with a "/" (e.g.: `/home/data/`) on Linux and "\" on Windows.
`scena`	integer	**Dust and solar EUV input scenario** : 1 = Climatology Scenario, solar EUV average conditions 2 = Climatology Scenario, solar EUV minimum conditions 3 = Climatology Scenario, solar EUV maximum conditions 4 = dust storm $\tau = 5$, solar minimum conditions 5 = dust storm $\tau = 5$, solar averaged conditions 6 = dust storm $\tau = 5$, solar maximum conditions 7 = warm scenario: dusty atmosphere, solar max 8 = cold scenario: low dust conditions, solar min. 24 = Mars Year 24, with associated solar EUV conditions. 25 = Mars Year 25, with associated solar EUV conditions. 26 = Mars Year 26, with associated solar EUV conditions. 27 = Mars Year 27, with associated solar EUV conditions. 28 = Mars Year 28, with associated solar EUV conditions. 29 = Mars Year 29, with associated solar EUV conditions. 30 = Mars Year 30, with associated solar EUV conditions. 31 = Mars Year 31, with associated solar EUV conditions. **N.B.** The solar conditions describe variations in the Extreme UV input which control the heating of the atmosphere above ~120 km, which typically varies on a 11 years cycle. The different dust scenarios differ by dust amount and distribution used to create the data files (dust is highly variable on Mars from year to year). **The "Climatology"** dust scenario is designed be representative of a baseline typical Mars year. **The warm and cold** scenario are provided to bracket the possible dust content of the atmosphere, outside global dust storms. **dust storm** $\tau = 5$ represents Mars during a global dust storm (dust opacity set to 5 and using a darker dust). Only available when such storms are likely to happen, during northern fall and winter (Ls=180-360). **MY24-MY31** scenarios are our best guess of the actual dust and EUV conditions for these eight Mars years. Please check the ***Detailed Design Document*** for further information.

Table 1: **CALL_MCD Input arguments (continued)**

`perturkey`	integer	Flag to set the type of perturbation to add. 1: None. 2: Add large scale perturbations (using EOFs). 3: Add small scale perturbations (gravity waves). 4: Add small and large scale perturbations. 5: Add `seedin` times the standard deviation. **N.B.** For the small scale or large scale perturbations, a seed for the random number generator must be specified (see `seedin` argument). When large scale perturbation is requested, as long as `seedin` remains the same, no new random vector is generated and you work with the same correlated perturbed atmosphere.
`seedin`	real	• if `perturkey=1`, `2`, `3` or `4`: Random number generator seed and flag. For the first call to `call_mcd`, this value (in fact its integer part) is used to seed the random number generator. If the value of `seedin` is changed between subsequent calls to `call_mcd`, it triggers the reseeding of the random number generator and subsequently the regeneration of a new perturbed atmosphere (see section 5.5.1). • if `perturkey=5`: coefficient by which the standard deviation should be multiplied before being added to the mean value. `seedin` is then not allowed to be more than 4 or less than -4.
`gwlength`	real	Wavelength of the vertical gravity wave (in meters). Used for small scale perturbations (ie: if `perturkey=3` or `4`). Should be between 2000 and 30000 m; if set to 0 then a default value of 16000 m is used. **N.B.** *Feature for specialists*: Changing the value of `gwlength` between calls to `call_mcd` triggers the generation of a new random phase for the gravity wave (and without altering the large scale perturbation, if the later is also requested, i.e. in the `perturkey=4` case).
`extvarkeys`	integer (dim 100)	**Flags to request extra variables on output** Each element i of this array signals wether the ith element of optional outputs (the `extvar` array) should be given on output. if `extvarkeys(i)=0`, then extra variable `extvar(i)` is not computed (note that the 7 first elements of output array `extvar` are always computed). if `extvarkeys(i)=1`, then extra variable `extvar(i)` is computed.

Table 2: **CALL_MCD output arguments**

Name	**Type**	**description**
`pres`	real	**Atmospheric pressure** (Pa)
`dens`	real	**Atmospheric density** (kg/m^3)
`temp`	real	**Atmospheric temperature** (K)
`zonwind`	real	**Zonal component of wind**, in m/s ($>$ 0 if eastward)
`merwind`	real	**Meridional component of wind**, in m/s ($>$ 0 if northward)
`meanvar`	real (dim 5)	**"mean" atmospheric values** • meanvar(1) = mean atmospheric pressure • meanvar(2) = mean atmospheric density • meanvar(3) = mean atmospheric temperature • meanvar(4) = mean zonal wind • meanvar(5) = mean meridional wind This array contains the unperturbed values of `pres`, `dens`, `temp`, `zonwind` and `merwind` (i.e.: In the `perturkey=1` case, where no perturbation are requested, the `meanvar` array will contain these).

Table 2: **CALL_MCD output arguments (continued)**

extvar	real (dim 100)	**Supplementary variables** array extvar(1) to extvar(7) provides time and space coordinate which are always computed and are therefore always set. Outputs extvar(8) to extvar(76) are only computed and set if the corresponding input argument `extvarkeys=(i)` is set to 1. These are otherwise set to zero. The rest of the array, extvar(77) to extvar(100) is unused (yet) and always set to zero. These available supplementary variables are: • extvar(1)= Radial distance to planet center (m). • extvar(2)= Altitude above areoid (Mars geoid) (m). • extvar(3)= Altitude above local surface (m). • extvar(4)= Orographic height (m) (altitude of the surface with respect to the areoid). **N.B.** Depending on the value of input flag `hireskey`, references to altitudes and orographic height are with respect to GCM grid or high resolution MOLA topography and areoid. • extvar(5)= Ls, solar longitude of Mars (deg). • extvar(6)= LTST: Local True Solar Time at longitude `lon` (in martian hours = 1/24 of a Mars day). • extvar(7)= Universal solar time (LTST at `lon`=0) (hrs). • extvar(8)= Cp: Air specific heat capacity (J.kg^{-1}.K^{-1}). • extvar(9)= $\gamma = Cp/Cv$ Ratio of specific heats. • extvar(10)= RMS day to day variations of density (kg/m^3). **N.B.** The given RMS is either pressure-wise (if `zkey`=4) or altitude-wise (if `zkey`=1, 2 or 3). • extvar(11)= LMT: Local Mean Solat Time at longitude `lon` (in martian hours = 1/24 of a Mars day). • extvar(12)= Sun-Mars distance (in Astronomical Unit AU) • extvar(13)= Scale height H (m) at given input altitude xz. • extvar(14)= GCM orography (m) (will be equal to extvar(4) if input parameter `hireskey`=0). **N.B.** Provided for specialist interested in the differences between low resolution (i.e.: the GCM resolution) and high resolution MOLA topography. • extvar(15)= Surface temperature (K). • extvar(16)= Daily maximum mean surface temperature (K). • extvar(17)= Daily minimum mean surface temperature (K). • extvar(18)= Surface temperature RMS day to day variations (K).

Table 2: **CALL_MCD output arguments (continued)**

		• extvar(19)= Surface pressure (Pa) (high resolution if `hireskey`=1, GCM surface pressure if `hireskey`=0).
		• extvar(20)= GCM surface pressure (Pa) (will be equal to `extvar(19)` if `hireskey`=0). **N.B.** Provided for specialist interested in the differences between low resolution (i.e.: the GCM resolution) and high resolution surface pressures.
		• extvar(21)= Atmospheric pressure RMS day to day variation (Pa), if `zkey`=1, 2 or 3. Otherwise set to zero.
		• extvar(22)= Surface pressure RMS day to day variations (Pa).
		• extvar(23)= Atmospheric temperature RMS day to day variations (K). **N.B.** The given RMS is either pressure-wise (if `zkey`=4) or altitude-wise (if `zkey`=1, 2 or 3).
		• extvar(24)= Zonal wind RMS day to day variations (m/s). **N.B.** The given RMS is either pressure-wise (if `zkey`=4) or altitude-wise (if `zkey`=1, 2 or 3).
		• extvar(25)= Meridional wind RMS day to day variations (m/s). **N.B.** The given RMS is either pressure-wise (if `zkey`=4) or altitude-wise (if `zkey`=1, 2 or 3).
		• extvar(26)= Vertical wind (m/s) (**positive when downward**).
		• extvar(27)= Vertical wind RMS day to day variations (m/s). **N.B.** The given RMS is either pressure-wise (if `zkey`=4) or altitude-wise (if `zkey`=1, 2 or 3).
		• extvar(28)= Small scale density perturbation (gravity wave) (kg/m^3).
		• extvar(29)= Surface roughness length z_0 (m).
		• extvar(30)= Solar flux reflected to space (W/m^2).
		• extvar(31)= Thermal IR ($\lambda > 5\mu$m) flux to surface (W/m^2).
		• extvar(32)= Solar flux ($\lambda < 5\,\mu$m) to surface (W/m^2).
		• extvar(33)= Thermal IR flux to space (W/m^2).
		• extvar(34)= Surface H_2O ice (seasonal frost) layer (kg/m^2). In areas covered by perrenial H_2O ice deposits, this seasonal frost layer is limited to maximum value 0.5 kg/m^2
		• extvar(35)= Surface CO_2 ice layer (kg/m^2).

Table 2: **CALL_MCD output arguments (continued)**

			• extvar(36)= DOD: Dust column visible optical depth. From local surface to the top of the atmosphere, at wavelength 0.67 μm
			• extvar(37)= Dust Optical Depth RMS day to day variations.
			• extvar(38)= Dust mass mixing ratio (kg/kg$_{air}$).
			• extvar(39)= Dust effective radius (m). Dust size distribution is assumed to follow a log-normal distribution with an effective variance of $\nu_{eff} = 0.5$. Therefore, the geometric mean radius r_0 can be derived from the effective radius r_{eff} as $r_0 = r_{eff} \times (1 + \nu_{eff})^{-5/2} = 0.36 \times r_{eff}$
			• extvar(40)= Dust deposition rate on a flat horizontal plane at the surface of Mars (kg.m^{-2}.s^{-1}).
			• extvar(41)= Water vapor column (kg/m^2). **N.B.** If you prefer to have this value in precipitable microns (pr-μm; i.e. g/m^2), then simply multiply it by 1000.
			• extvar(42)= Water vapor vol. mixing ratio (mol/mol).
			• extvar(43)= Water ice column (kg/m^2).
			• extvar(44)= Water ice mixing ratio (mol/mol).
			• extvar(45)= Water ice effective radius (m). Water ice cloud particule size distribution is assumed to follow a log-normal distribution with an effective variance of $\nu_{eff} = 0.1$. Therefore, the geometric mean radius r_0 can be derived from the effective radius r_{eff} as $r_0 = r_{eff} \times (1 + \nu_{eff})^{-5/2} = 0.8 \times r_{eff}$
			• extvar(46)= Convective planetary boundary layer (PBL) height (m).
			• extvar(47)= Maximum upward convective wind (m/s) within the planetary boundary layer (PBL).
			• extvar(48)= Maximum downward convective wind (m/s) within the planetary boundary layer (PBL).
			• extvar(49)= Convective vertical wind variance (m^2.s^{-2}) at input altitude xz. This quantity has only a meaning inside the PBL; it is set to zero if the sought input xz altitude is above the PBL.
			• extvar(50)= Convective eddy vertical heat flux (m.s^{-1}.K^{-1}) at input altitude xz. This quantity has only a meaning inside the PBL; it is set to zero if the sought input xz altitude is above the PBL.
			• extvar(51)= Suface wind stress (kg.m^{-1}.s^{-2}).
			• extvar(52)= Surface sensible heat flux (W.m^{-2}). Negative when the flux is from the surface to the atmosphere
			• extvar(53)= R : Reduced molecular gas constant (J.kg^{-1}.K^{-1}) of the atmosphere at input altitude xz.
			• extvar(54)= Air viscosity v estimation (N.s.m^{-2}).

Table 2: **CALL_MCD output arguments (continued)**

		• extvar(55)= Not used (set to zero). • extvar(56)= Solar zenith angle (degrees). • extvar(57)= CO_2 volume mixing ratio (mol/mol$_{air}$). • extvar(58)= N_2 volume mixing ratio (mol/mol$_{air}$). • extvar(59)= Ar volume mixing ratio (mol/mol$_{air}$). • extvar(60)= CO volume mixing ratio (mol/mol$_{air}$). • extvar(61)= O volume mixing ratio (mol/mol$_{air}$). • extvar(62)= O_2 volume mixing ratio (mol/mol$_{air}$). • extvar(63)= O_3 volume mixing ratio (mol/mol$_{air}$). • extvar(64)= H volume mixing ratio (mol/mol$_{air}$). • extvar(65)= H_2 volume mixing ratio (mol/mol$_{air}$). • extvar(66)= electron number density (cm^{-3}). values are only given for pressures higher than $5\ 10^{-6}$ Pa (roughly up to 200 km above the surface); above this ”chemistry ionosphere”, one would need to model the full ionosphere dynamics, which isn't the case here. • extvar(67)= CO_2 column (kg.m^{-2}). • extvar(68)= N_2 column (kg.m^{-2}). • extvar(69)= Ar column (kg.m^{-2}). • extvar(70)= CO column (kg.m^{-2}). • extvar(71)= O column (kg.m^{-2}). • extvar(72)= O_2 column (kg.m^{-2}). • extvar(73)= O_3 column (kg.m^{-2}). • extvar(74)= H column (kg.m^{-2}). • extvar(75)= H_2 column (kg.m^{-2}). • extvar(76)= Total Electronic Content (TEC) (m^{-2}). As computed using electron content from the ”chemistry ionosphere”, i.e. from the surface to $5\ 10^{-6}$ Pa. • extvar(77) to extvar(100): Not used (set to zero).
`seedout`	real	The current index of the random number generator. May be used to trigger (by setting `seedin` to this value) the generation of a new set of perturbations for the next call to `call_mcd`.

Table 2: **CALL_MCD output arguments (continued)**

`ier`	integer	**Status code**: When an error occurs in `call_mcd`, all the outputs arguments (`pres`, `dens`, `temp`, `zonwind`, `merwind`, all elements of `meanvar` and `extvar`)are set to −999 and a message is written to the standard output. The value of `ier` summarises the status of `call_mcd`: 0: OK (no error) 1: Wrong vertical coordinate flag `zkey` 2: Wrong choice of dust scenarion `scena` 3: Wrong value for perturbation flag `perturkey` 4: Wrong value for high resolution flag `hireskey` 5: Wrong value for date flag `datekey` 6: Wrong value for extra variables output flag `extvarkey` 7: Wrong value for latitude `xlat` 8: Inadequate value for gravity wave wavelength `gwlength` 9: Wrong value of input solar longitude (`xdate` must be in [0:360], in the `datekey`=1 case) 10: Given Julian date `xdate` (in the `datekey`=0 case) implies an Earth date outside of [1800:2200] range 11: Wrong value of local time (in the `datekey`=1 case), which should be in [0:24] 12: Incompatible `localtime`≠0 and `datekey`=0 13: Unresonable value of `seedin` (in `perturkey`=5 case) 14: No dust storm scenario available at such date 15: Could not open a database file (`dset` is probably wrong) 16: Failed loading data from a database file 17: Sought altitude is underground 18: adding (`perturkey`=5) perturbation yields unphysical density 19: adding (`perturkey`=5) perturbation yields unphysical temperature 20: adding (`perturkey`=5) perturbation yields unphysical pressure

5.5 The right use of the CALL_MCD subroutine

5.5.1 Perturbed atmospheres

In addition to the mean atmospheric state, the user may obtain a realisticly perturbed (using input flag `perturkey`) atmosphere.

The **perturbation consisting of adding n times the standard deviation to the mean value** (the `perturkey = 5` case) must not be used to create randomly perturbed atmospheres, but only as a mean to globally overestimate or underestimate the profiles of the meteorological variables. To generate randomly perturbed atmospheres you must use small or large scale perturbations which

Table 3: Example of Fortran code to illustrate the use of (re-)setting perturbations

```
! build a density profile at a given time and location
! with EOF and GW perturbations
        perturkey=4
        seedin=100   ! seed perturbations
        zkey=3  ! work in "altitude above local surface" coordinate
        do i=1,100
          xz=(i-1)*2000.0 ! go from surface to 200km
          call_mcd(zkey,xz,xlon,xlat,hireskey,
     &            datekey,xdate,localtime,dset,scena,
     &            perturkey,seedin,gwlength,extvarkeys,
     &            pres,dens,temp,zonwind,merwind,
     &            meanvar,extvar,seedout,ier)
          profile(1,i)=dens ! store density
        enddo
!
! ... some code here...
! ... moved on to a different time or place 'far' from previous one
! ... such that perturbations should be reset
!
      seedin=seedout  ! change seedin to regenerate perturbations
      ! build the new density profile
        do i=1,100
          xz=(i-1)*2000.0 ! go from surface to 200km
          call_mcd(zkey,xz,xlon,xlat,hireskey,
     &            datekey,xdate,localtime,dset,scena,
     &            perturkey,seedin,gwlength,extvarkeys,
     &            pres,dens,temp,zonwind,merwind,
     &            meanvar,extvar,seedout,ier)
          profile(2,i)=dens ! store density
        enddo
```

take into account some correlation of perturbations in the space and between variables. Then when you use the `perturkey = 5` perturbation, you have to keep the same `seedin` (ie: multiplying factor) along the whole trajectory to avoid introducing unrealistic gradients between consecutive values.

When generating a randomly perturbed atmosphere **using the large scale perturbations** (`perturkey = 2` or `4`) to simulate a trajectory, the value of `seedin` should be kept constant in order to work with the same correlated perturbed atmosphere.
Reseting the large scale perturbations (by modifying the value of `seedin` between calls to `call_mcd`) to build profiles at a given location should only be done in the context of generating a range of different possibilities (an example is given in table 3).

When **using the perturbation due to gravity wave propagation (small scale perturbation)** (`preturkey = 3` or `4`), to generate a vertical profile, the phase (ie: the value of `seedin`), as well as the associated wavelength `gwlength` should remain fixed.

It must be emphasized that the gravity wave perturbation scheme is purely vertical and does not include any horizontal component or coherence:

- When computing a group of trajectories clustered over a small horizontal distance (and at a given martian time), then perturbations should not be reset between the computation of each trajectory in order to retain the underlying correlation of the perturbations.

- When computing a trajectory over large horizontal distance (e.g. aerobraking), the effect of travelling through non-coherent gravity waves can be mimicked by changing the seed about every 50 km. This will create "density wall" however. This will improved in the next version of the Mars Climate Database.

Note for specialists: It has been made possible to keep a given large scale perturbation phase whilst only changing the gravity wave small scale perturbation between calls. This is achieved by changing the value of input argument `gwlength` (the gravity wave wavelength) between calls to `call_mcd`. As mentionned above, this feature, which should be usefull to people who might want to generate "realistic" longitude-height or latitude-height slices over large horizontal distances (over which indeed gravity waves should not correlate, as mentionned above).

5.5.2 Running time

In order to minimize computational time, the datasets corresponding to encompassing months (of sought input date) of a given MCD dust and EUV scenario are loaded from the database at the first call of the '*call_mcd*' subroutine. This initial loading is time consuming, but once loaded these values can then be used for further calls, as long as the sought dates do not lead to a change in bracketing months. This should be taken into account when simulating trajectories (or maps) over more than a month; calls to `call_mcd` should be ordered so that all data is gathered within each 30° range of Ls (15 to 45, 45 to 75, ...).

Similarly, only required data sets are loaded: if for instance no perturbations are requested, then only mean values are loaded. Note that requesting perturbations (`perturkey` set to 2,3 or 4), as well as supplementary outputs (elements of `extvarkeys` set to 1), implies extra computations which will slow down the program.

6 Calling the CALL_MCD subroutine from IDL

The Interactive Data Language (IDL) is a commercial software for data analysis and visualization tool that is widely used in earth, planetary science and astronomy.

The `mcd/idl` subdirectory contains two tools which show how the `call_mcd` subroutine may be called from IDL. Note that `call_mcd` is not directly called from IDL as such, auxiliary Fortran programs are used. These programs are launched from the IDL session, and their output (written to an intermediate file) is then loaded and used.

- **mcd_idl.pro** is an IDL procedure that can be used to retrieve a block of atmospheric data from the Mars Climate database.
  - **inputs:** Solar longitude Ls (in degrees), Local time Loct (in martian hours), latitude lat (degrees north), east longitude lon (degrees),dust scenario dust (from 1 to 8),vertical coordinate type zkey (1, 2, 3, or 4), high resolution mode hireskey (on:1, off:0) and vertical coordinate xz (altitude, m or pressure level, Pa). *All the time and space coordinate can be IDL arrays*. For instance, if you want to make a map of temperature at a given local time, altitude, and Ls, then in input lat and lon should be arrays.
  - **outputs:** "meanvarz" and "extvarz" as defined for `call_mcd`, *except that they are multidimensional arrays depending on the dimension of the inputs*. For instance, in our example of a map, the IDL dimension of meanvarz will be DBLARR[5+1,nlat,nlon] [5]. "5+1" means here that contrary to the usual IDL convention meanvarz and extvarz left index is only used starting at 1 to keep the same numbering than in the Fortran code and thus follow the "`call_mcd` subroutine outputs" given in section 5.4.

[5]When multidimensinal data is sought, dimensions are arranged as follows: meanvarz[6,nlat,nlon,nz,nLs,nloctime]

  - **How to use it:**
    1) Copy (you may also just use symbolic links) files `constants_mcd.inc`, `call_mcd` and `heights.F` from parent dircetory `mcd/` to the current directory.
    2) Edit the the Fortran code `mcd_idl.F` (which is used by IDL to call `call_mcd`) and set the path (the `dset` variable) to the MCD data directory.
    3) Compile the Fortran code `mcd_idl.F`. You may use the script `compile_mcd_idl` (after having edited it to match your needs).
    4) Call `mcd_idl.pro` from IDL using your favorite method: File `test_idl.pro` is provided as an example of an IDL program which uses `mcd_idl.pro`.

- **profils_mcd_idl.pro** is an IDL procedure that can be used to retrieve atmospheric profiles for a list of horizontal coordinates and times. This is especially useful to emulate observations of atmospheric profiles from an instrument.

  - **inputs:** Solar longitude Ls (in degrees), Local time Loct (in martian hours), latitude lat (degrees north), east longitude lon (degrees),dust scenario dust (from 1 to 8),vertical coordinate type zkey (1, 2, 3, or 4), high resolution mode hireskey (on:1, off:0) and vertical coordinate xz (altitude, m or pressure level, Pa). *xz is the vector of altitude (m) or pressure (Pa) defining the vertical coordinates of the profiles. Ls, Local time, lat and lon must be array of the same size, providing a list of coordinate where you want to retrieve profiles.*
  - **outputs:** "meanvarz" and "extvarz" as defined for `call_mcd`, *except that they are arrays containing the profiles for the list of horizontal and time coordinates.* For instance "meanvarz" has the following dimension: DBLARR(5+1,number of profiles,number of point in each profile). "5+1" means here that contrary to the usual IDL convention meanvarz and extvarz left index is only used starting at 1 to keep the same numbering than in the Fortran code and thus follow the "`call_mcd` subroutine outputs" given in section 5.4.
  - **How to use it:**
    1) Copy (you may also just use symbolic links) files `constants_mcd.inc`, `call_mcd` and `heights.F` from parent dircetory `mcd/` to the current directory.
    2) Edit the the Fortran code `profils_mcd_idl.F` which is used by IDL to call `call_mcd` and set the path (the `dset` variable) to the MCD data directory.
    3) Compile the Fortran code `profils_mcd_idl.F`. You may use the script `compile_profils_mcd_idl` (after having edited it to match your needs).
    4) Call `profils_mcd_idl.pro` from IDL using your favorite method: the file `test_profils_idl.pro` is an example of an IDL program which uses `profils_mcd_idl.pro`.

7 Calling the CALL_MCD subroutine from Matlab

The IDL tools described above, have then been translated into similar matlab scripts (initially for MCDv4.0 by Kerri Kusza, Stanford University) which are available in directory `mcd/matlab`.

As with the IDL interfaces, data is retrieved from the database via auxiliary Fortran programs. Function `mcd_mat.m` (along with `mcd_mat.F`) can be used to retreive a block of atmospheric data, and function `profils_mcd_mat.m` (along with `profils_mcd_mat.F`) can be used to retrieve a vertical profile of atmospheric data. See the `README` file in the same directory for further comments on adapting these interfaces to your settings.

8 Calling the CALL_MCD subroutine from Scilab

Scilab is a free open source scientific software (similar to Matlab) providing a powerful computing environment for engineering and scientific applications.

Examples of tools similar to those mentionned above, and adapted to Scilab by Aymeric Spiga, are available in directory `mcd/scilab`. As for the Matlab and IDL interfaces described previously, data is retreived from the database via auxiliary Fortran programs which read and write their inputs and outputs to and from intermediate files. The function defined in file `mcd.sci` (which calls the Fortran program `mcd_sci`, obtained by compiling `mcd_sci.F`) is usefull for retreiving a block of atmospheric data and the function defined in file `profils_mcd.sci` (which calls program `profils_mcd_sci` obtained by compiling Fortran source code `profils_mcd_sci.F`) can be used to retreive a vertical profileof atmospheric data. There is moreover a short script `mcd_plot.sci` which illustrates the use of these functions. Once the Fortran routines adapted and compiled (see the `README`, `compile_mcd_sci` and `compile_profils_sci` files), this demo may be lauched with the following command:

```
scilab -f mcd_plot.sci
```

9 Calling the CALL_MCD subroutine from C or C++ programs

Examples of C and C++ programs interfaced with the Fortran subroutine `call_mcd` are given in the `mcd/c_interfaces` subdirectory. These files, along with the header file `mcd.h`, illustrate how to call the Fortran subroutines `call_mcd` and `julian` from C (`test_emcd.c`) and C++ (`test_emcd.cpp`) main programs.

Unfortunately, inter-language calling conventions vary with compilers and operating systems; although the C and C++ interfaces have been tested on our Linux systems, using Gnu compilers (gfortran, gcc, g++) as well as Portland Group compilers (pgfortran, pgcc, pgCC), they will certainly need to be adapted to your settings. Some examples of compiling and linking commands are given in the provided `Makefile` and `README` files.

To summarize, compiling and linking in order to build the main (C or C++) program requires the following steps:

1. Create the object files corresponding to the Fortran subroutines which will be called by the main program, e.g.:
   ```
   > pgf90 -c julian.F
   > pgf90 -c heights.F -I/path/to/netcdf/include
   > pgf90 -c call_mcd.F -I/path/to/netcdf/include
   ```
   which will create objects `julian.o`, `call_mcd.o` and `heights.o`.

2. Compile the main program (e.g. `test_mcd.c`) with your C or C++ compiler:
   ```
   > pgcc test_mcd.c call_mcd.o julian.o heights.o
   -I/path/to/netcdf/include -L/path/to/netcdf/lib
   -l netcdf
   ```
   you will likely need to add other libraries to ensure good Fortran/C/C++ compatibility, but these are extremely compiler (and platform) dependent; check your compiler's manual for instructions.

10 Calling the CALL_MCD subroutine from python

An exemple of a python script calling the Fortran subroutine `call_mcd` is given in the `mcd/python` subdirectory.

Note that in order to interface the MCD software with python, one must first create the corresponding python interface, using the provided `fmcd_gfortran.sh` script, adequately adapted as explained in the `README` file in the directory.

The provided `test_mcd.py` script illustrates how one can implement a call to the MCD from python. Users interested in more advanced interfacing (e.g. using the MCD as a python class) should take a look at: `https://github.com/aymeric-spiga/mcd-python`

11 High accuracy surface pressure tool `pres0`

The subdirectory `mcd/pres0` contains a tool specifically designed to compute surface pressure (as well as surface altitude above the areoid) at any location and time on Mars (outside global dust storm; data corresponding to the Climatology scenario is used), with the best accuracy currently possible.

As of version 4.2 of the Mars Climate Database, this feature (and its extention to atmospheric variables) is included in the `call_mcd` access software (by setting input argument `hireskey=1`). The now redundant `pres0` tool is nonetheless kept as it is a convenient light and autonomous[6] tool for users who only need to retreive high resolution topography and surface pressure.

More information on how `pres0` works is available in the **Detailed Design Document**.

11.1 How to use `pres0`

See the `README` file in subdirectory `mcd/pres0`. This directory also contains :

- the file `pres0.F` ,which contains the `pres0` main subroutine and the subroutines it calls and uses
- The file `testpres0.F` which contains a simple example of a program calling `pres0`.
- The file `compile` which contains an example of a simple command to compile the programs

11.2 Input/output of subroutine `pres0`

A call to `pres0` should be as follows:

```
call pres0(dset,lat,lon,solar,loctim,pres,alt,ierr)
```

- Pres0 needs 5 input arguments:
  - `dset` (`character*(*)`): Path to datafiles `VL1.ls`, `mola_32.nc` and `ps_MY24.nc`. These are in the same directory as all the database datafiles[7]. The `dset` string must end with a '`/`'.
  - `lat` (`real`): Latitude coordinate of the point (in degrees North).
  - `lon` (`real`): Longitude coordinate of the point (in degrees East).
  - `solar` (`real`): Solar longitude Ls (in degrees).
  - `loctim` (`real`): Local time (in martian hours).
- And fills 3 output values:
  - `pres` (`real`): Surface pressure (Pa) at given space and time coordinates.
  - `alt` (`real`): Above areoid altitude of the surface (m) at given space coordinates.
  - `ierr` (`integer`): control variable (0 if all is ok).

[6]The only files that pres0 requires are `VL1.ls`, `mola32.nc` and `ps_clim.nc`. Users interested in installing only pres0 and not the whole database, should copy these files (which are in the `data` directory) to a local directory.

[7]We recomend using the same symbolic link stategy as given in section 3.2.

12 The `heights` tool

The `call_mcd` routine handles and converts various types of vertical coordinates, as explained in section 5.4. Users interested in having a light and fast tool for converting vertical coordinates expressed as distance to the center of the planet, height above the areoid (zero datum) and height above the local surface may use the `heights` subroutine. Given any of the three, this routine computes the other two.

Just as `call_mcd`, `heights` can work in high resolution (i.e.: using high resolution 32 pixels/degree MOLA topography and areoid) or low resolution (i.e.: at GCM horizontal grid resolution of 5.625×3.75) mode. At GCM resolution, topography and areoid are read from the `mountain.nc` datafile. At high resolution, the MOLA topography file `mola32.nc` and spherical harmonics expansion coefficients (in file `mgm1025`) are used. All these files are stored in the `data` directory of the MCD distribution.

12.1 Arguments of `heights` subroutine

A Fortran call to the `heights` subroutine should be as follows:

```
      call heights(dset,xlat,xlon,hireskey,convkey,
     &                    zradius,zareoid,zsurface,ier)
```

where input and output arguments are:

- `dset (character*(*))`: Path to the datafiles the routine needs. If left empty (e.g.: `dset=' '`) the default path `'MCD_DATA/'` is assumed.
- `xlon (real)`: East planetocentric longitude (in degrees).
- `xlat (real)`: North planetocentric latitude (in degrees).
- `hireskey (integer)`: Flag to set the resolution (0: GCM resolution, 1: high resolution)
- `convkey (integer)`: Switch to indicate which distance is known and used to find the other two:
  `convkey=1`: `zradius` is known, compute `zareoid` and `zsurface`.
  `convkey=2`: `zareoid` is known, compute `zradius` and `zsurface`.
  `convkey=3`: `zsurface` is known, compute `zradius` and `zareoid`.
- `zradius (real)`: distance to center of planet (m).
- `zareoid (real)`: altitude above areoid (m).
- `zsurface (real)`: altitude above local surface (m).
- `ier (integer)`: Routine status/error code (=0 if all went well, see file `heights.F`).

A Differences between Version 5.2 and previous versions of the MCD

- Previous Mars Climate Database users are warned that the definition, the units, and the numbering in the `extvar` array) of variables can vary between versions.
  Main changes between version 5.2 and version 5.1 are the following:

  - Added scenarios 24 to 31 for Mars Years 24 to 31.
  - Vertical velocity and associated RMS (`extvar[26]` and `extvar[27]`) have been corrected.
  - Total electronic content (TEC, `extvar[76]`) is better evaluated (using ”chemistry ionosphere” electronic content only).
  - The interpolation scheme for the near-surface layer (first few meters above the surface) has been improved.
  - More control when using combined small and large scale perturbation schemes to avoid (rare cases of) unphysically negative density.
  - Added Local Mean Solar Time to outputs (`extvar[11]`)
  - Added Sun-Mars distance to outputs (`extvar[12]`)
  - Added solar zenith angle to outputs (`extvar[56]`)

- **The differences in the design and the content of the Mars Climate Database are detailed in the MCD 5.2 Detailed Design Document, Appendix B.**